NSI 1ère - Linux et les systèmes UNIX

QK

Linux et les systèmes libres

Linux est le noyau d'un système d'exploitation (OS) dont **debian** est une distribution.

- C'est un système d'exploitation libre dont il existe de nombreuses variantes appelées *distributions* fonctionnant elles-mêmes sur le noyau linux.
- Linux désigné à la fois le *noyau (kernel)* qui lance la machine et gère le bas niveau (proche du métal) et le système d'exploitation lui même (les applications).
- Linux est un dérivé d'UNIX, système crée dans les années 70, codé en langage C (crée à cette occasion). Ces deux "logiciels" : C et UNIX constituent à la fois le langage et le système les plus importants de l'histoire informatique (la majorité des langages actuels dérivent du C et la majorité des machines professionnelles tournent sur un dérivé d'UNIX).
- UNIX et ses dérivés sont présents partout : tous les smartphones fonctionnent sur un de ses dérivés (Linux pour android, bsd pour iOS) ainsi que les mac.
- UNIX fonctionne sur n'importe quelle machine : une caméra IP, un super ordinateur (les 500 plus puissants utilisent tous linux), un chromecast, une switch etc.
- Il existe des versions payantes de Linux (pour les professionnels), **debian** est un système d'exploitation gratuit et reconnu par les professionnels pour sa grande stabilité. Il n'est pas rare de rencontrer un serveur debian fonctionnant sans redémarrer depuis plusieurs années :

  Dernière coupure de courant chez moi ? Il y a 147 jours :

  ```
  $ uptime
  08:05:15 up 147 days, 23:35,  1 user,  load average: 0.52, 0.38, 0.37
  ```

Usage courant de linux

Hormis pour des logiciels très spécifiques, utiliser windows, linux ou OSX ne change rien. La grande majorité des logiciels existent sur toutes les plateformes hormis quelques exceptions notables :

- Photoshop et la suite adobe : windows et osx. De nombreuses alternatives
- les jeux vidéos : windows (**possible** de jouer sous linux ou osx mais demande du courage).

Grandes familles de systèmes d'exploitation

On en rencontre massivement deux :

- Windows et ses dérivés (MSDOS (~1985), Windows NT (1999), windows 7->10 (2008)). Domine le marché du PC "personnel"
- UNIX et ses dérivés : bsd (systèmes embarqués, réseaux d'entreprises), linux (partout dont android, super calculateurs, PC personnels, serveur web), OSX & iOS (produits apple)

Utilisation d'un système UNIX

Les trois couches d'UNIX

- Le noyau (kernel) : proche du métal. Lance la machine, gère la carte graphique, le réseau etc.
- la coquille (shell) : programme qui permet d'exécuter des utilitaires et d'intéragir (via les fenêtres ou le terminal)
- les utilitaires (ls, firefox) : les programmes qu'on fait tourner grâce au shell

Deux modes d'utilisation :

- graphique (GUI : graphical user interface)
- terminal (CLI : command line interface)

Le premier vous connaissez : fenêtres, clic clic clic.

Le second repose sur la console. On est devant un terminal qui exécute une boucle REPL (read eval print loop) :

1. On tape une commande. Le shell la lit (read)
2. Le système traite la commande et calcule une réponse (eval)
3. Il affiche une réponse (print)
4. On recommence (loop)

Dans un extrait de code on symbolise une commande shell par un `$` (souvent appelé *prompt*)

Dans un terminal on tape des commandes, elles sont exécutées par le shell. Celui de linux est souvent `bash`. `bash` est un langage de programmation interprété. Vous pouvez aussi créer des scripts bash (~scripts python moins faciles à écrire). `bash` est présent sous windows. `OSX` utilise `zsh`.

```
$ pwd
/home/quentin/travail
$ ls
cours.pdf   notes.csv  devoir.pdf
$ cd ..
$ ls
travail    personnel
```

On navige avec `cd`, on affiche les contenus avec `ls`, on se repère avec `pwd`

On peut accéder à un terminal physiquement (devant la machine) en lançant un programme ou à distance grâce au service réseau `ssh` (secured shell). C'est ainsi que sont administrées la majorité des machines 'professionnelles'.

Arborescence UNIX

Tous les systèmes UNIX accordent une grande importance aux fichiers.

UNIX VOIT SES PÉRIPHÉRIQUES ET SES PROCESSUS COMME DES FICHIERS.

Par exemple :

```
$ ls /
bin   dev         lib    media  proc  run   sys  var
boot  etc         lib64  mnt    root  sbin  tmp
home  lost+found  opt    srv    usr
```

- `/` le dossier racine de l'arborescence
- Tous les dossiers qui se terminent par `bin` contiennent des exécutables en binaire.
- `home` : dossiers des utilisateurs
- `dev` : le matériel
- `etc` : les reglages
- `root` : dossier de l'utilisateur root
- `mnt` et `media` : les *points de montage* des disques et partitions externes (là où apparaissent les clés usb etc.)

Permissions

La sécurité sous unix est gérée par la notion de permission.

- Un utilisateur ne peut pas faire ce qu'il veut.
- Le super utilisteur `root` peut tout faire. Devenir `root` avec `$ su`, exécuter une commande comme root avec `$ sudo commande`.
- Pourquoi ? Parce qu'en tant que `root` il suffit de 8 caractères pour effacer tous les disques de la machine...

L'affichage détaillé d'un fichier (`ls -lah`) montre les permissions de

- l'utilisateur courant
- de son groupe
- et de tout le monde

Exemple :

```
-rwxr-xr-x 1 quentin quentin  324  2 déc.  21:45 deploy.sh
-rw-r--r-- 1 quentin quentin 3,6M  5 déc.  08:32 inside.log
```

traduction

```
- : pas activé
d : directory
r : droit de lecture
w : droit d'écritude
x : droit d'exécution
```

- `deploy.sh` : ce n'est pas un dossier, je peux lire et écrire dans le fichier, l'exécuter. Mon groupe ne peux pas écrire dedans, les autres non plus.
- `inside.log` : tout le monde peut le lire, je suis le seul à pouvoir y écrire.

On change les permissions avec `chmod`

- soit en ajoutant ou retirant un flag : `$ chmod +x inside.log` rendra ce fichier exécutable
- soit en décrivant la permission par 3 nombres : `chmod 124 inside.log`
  - 1 : je peux exécuter
  - 2 : mon groupe peut écrire
  - 4 : tlm peut lire

  On fait la somme des nombres qu'on veut activer : Ex $7 = 1 + 2 + 4$ = tous les droits.

  `$ chmod 764 inside.log` je peux tout faire, mon groupe ne peut pas exécuter, tlm peut lire.

Processus

- Programme : du texte, décrivant des opérations à la machine.
- processus : un programme en cours d'exécution par la machine.

Chaque fois qu'on lance un programme, UNIX crée un processus.

On accède aux processus avec `ps`

```
$ ps -ef | grep Python
quentin  26945 10317  0 08:32 pts/1    00:00:00 /usr/bin/python -O /usr/bin/ranger
```

Affiche tous les processus, dans une table et filtre pour ne garder que ceux qui font référence à Python.

En gros, les programmes Python qui tournent sur la machine.

Je fais tourner un programme appelé `ranger` (gestionnaire de fichiers). Son numéro (PID) est 26945. Le numéro de son parent est 10317. Si ranger a planté et que je veux le tuer :

`$ kill 26945` ou `$ killall ranger`

Gestionnaire de ressources `$ top`

Redirection des entrées sorties

UNIX fonctionne principalement avec des petits programmes exécutant quelques tâches simples, le plus souvent une seule.

Ils communiquent avec des flux de texte qu'on peut enchaîner ou rediriger.

On peut, par exemple :

- écrire dans un fichier `$ cat cours.txt >> bonjour.txt` va recopier cours.txt dans bonjour.txt
- rediriger : `$ ps -ef` affiche 20 pages... `$ ps -ef | less` les fait défiler une par une !

Les rudiments du réseau

Afficher l'état du réseau.

```
$ ip a
$ ifconfig
```

Puis-je joindre une machine ?

```
$ ping 192.168.1.1
$ ping google.com
```

CTRL+C pour arrêter